

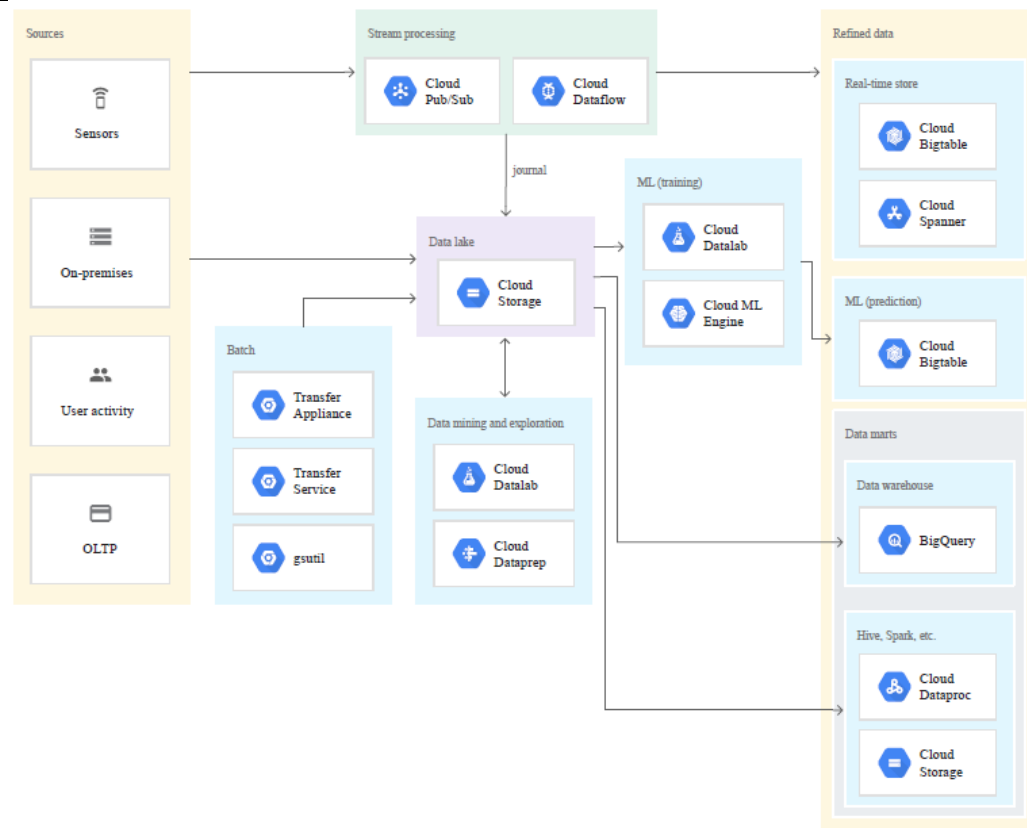
Exhibit A

'092 Patent

Infringement Chart for US 10,154,092 vs. Google**Claim 13**

Claim Language	Google Evidence
<p>13. A method comprising:</p>	<p>Google Storage Transfer Service is a service(method) for transferring data from cloud or on-premises sources</p> <p>What is Storage Transfer Service?</p> <p>Storage Transfer Service is a product that enables you to:</p> <ul style="list-style-type: none"> • Move or backup data to a Cloud Storage bucket either from other cloud storage providers or from your <u>on-premises</u> (/storage-transfer/docs/key-terms#on-prem) storage. • Move data from one Cloud Storage bucket to another, so that it is available to different groups of users or applications. • Periodically move data as part of a data processing pipeline or analytical workflow. <p>Overview Cloud Storage Transfer Service Documentation Google Cloud Source: https://cloud.google.com/storage-transfer/docs/overview</p>
<p>receiving input/output (I/O) traffic from a host device via a dedicated</p>	<p>Google Storage Transfer Service input/output (I/O) traffic from a host device (on-premises sources)</p>

I/O channel at a first interface, the I/O traffic comprising a write command;

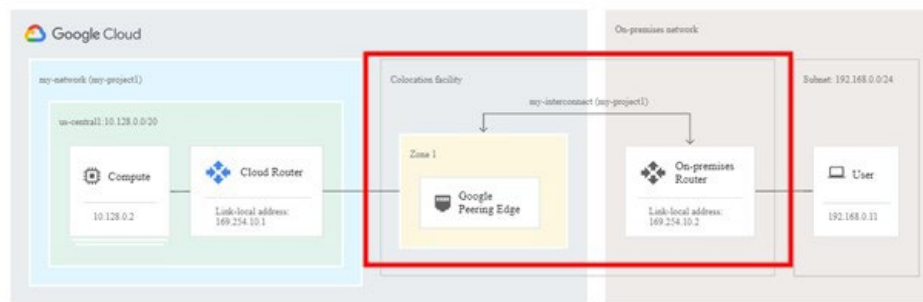


Cloud Storage as a data lake | Cloud Architecture Center | Google Cloud
 Source: <https://cloud.google.com/architecture/build-a-data-lake-on-gcp>

traffic from a host device (on-premises sources) via a dedicated I/O channel (Dedicated Interconnect Work) at a first interface.

How does Dedicated Interconnect work?

For Dedicated Interconnect, you provision a cross connect between the Google network and your own router in a common location. The following example shows a single Dedicated Interconnect connection between a GCP VPC network and on-premises network:

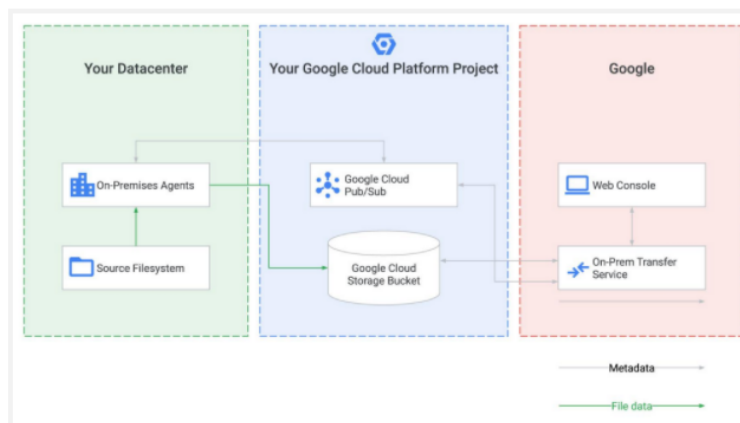


Dedicated Interconnect overview

Source: <https://cloud.google.com/storage-transfer/docs/on-prem-overview>

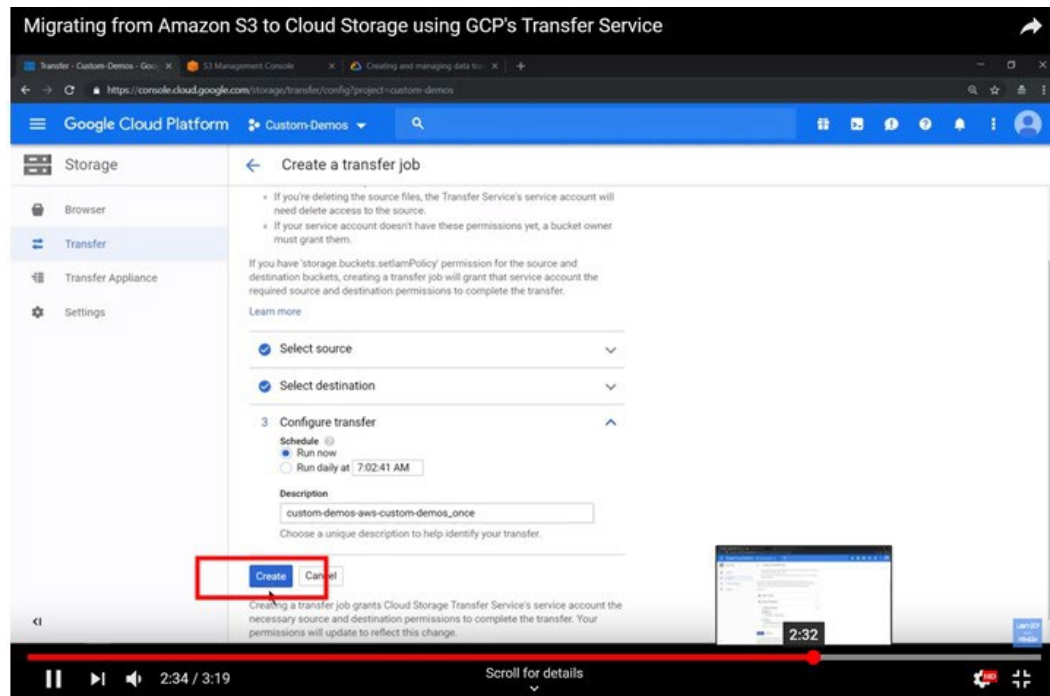
Getting started with Transfer Service for on-premises data

Here's how it works. First, install and start the on-premises software (the agent), then go to the Cloud Console and submit directories to transfer to Cloud Storage. When transferring data, the service will parallelize your transfer across many agents, and then coordinate these agents to transfer your data over a secure internet connection to Cloud Storage. Transfer Service for on-premises data also features a fully self-service GUI with detailed transfer logs so that you can create, monitor, and manage transfer jobs with confidence.

**Introducing Storage Transfer Service for on-premises data**

Source: <https://cloud.google.com/blog/products/storage-data-transfer/introducing-storage-transfer-service-for-on-premises-data>

When user clicks on the create button highlighted with red rectangle in the figure 2. An I/O command is received by the processor of allocated cloud machine to transfer data from any other cloud to a bucket of google platform cloud, which appears to relate to “receiving input/output (I/O)” the purpose of the create command is to load files into google cloud storage bucket, which appears to relate to write command.

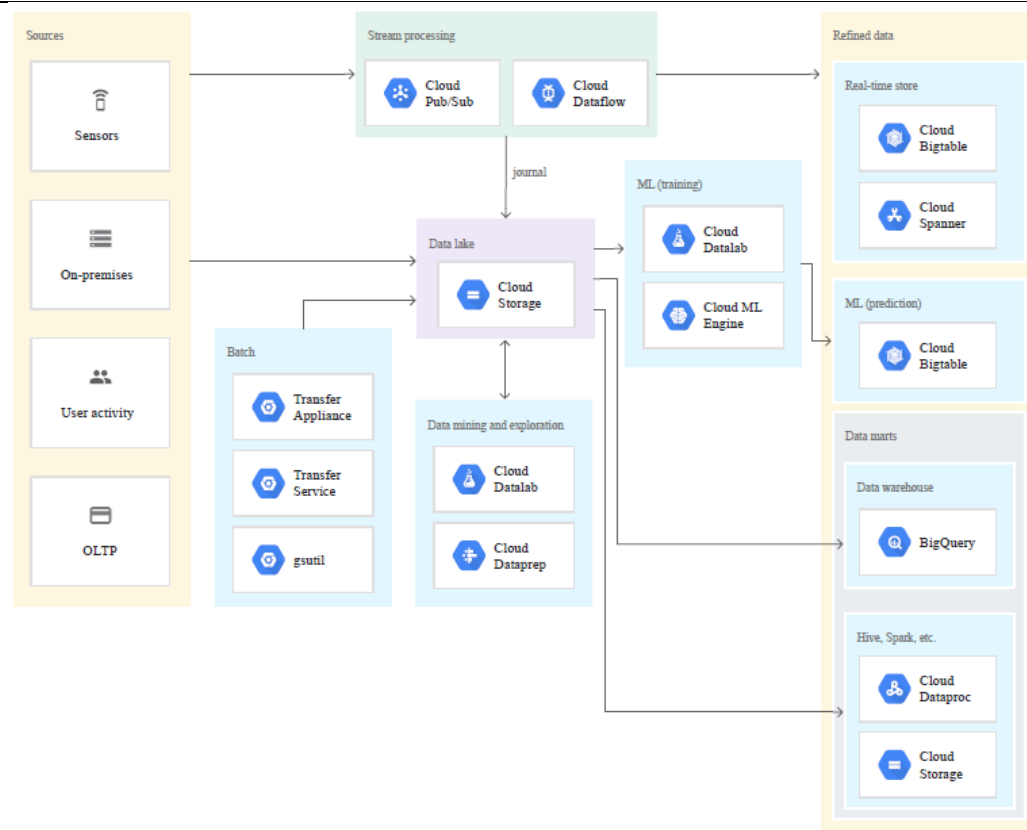


Source: <https://youtu.be/sBNtvG5D2CM?t=154>

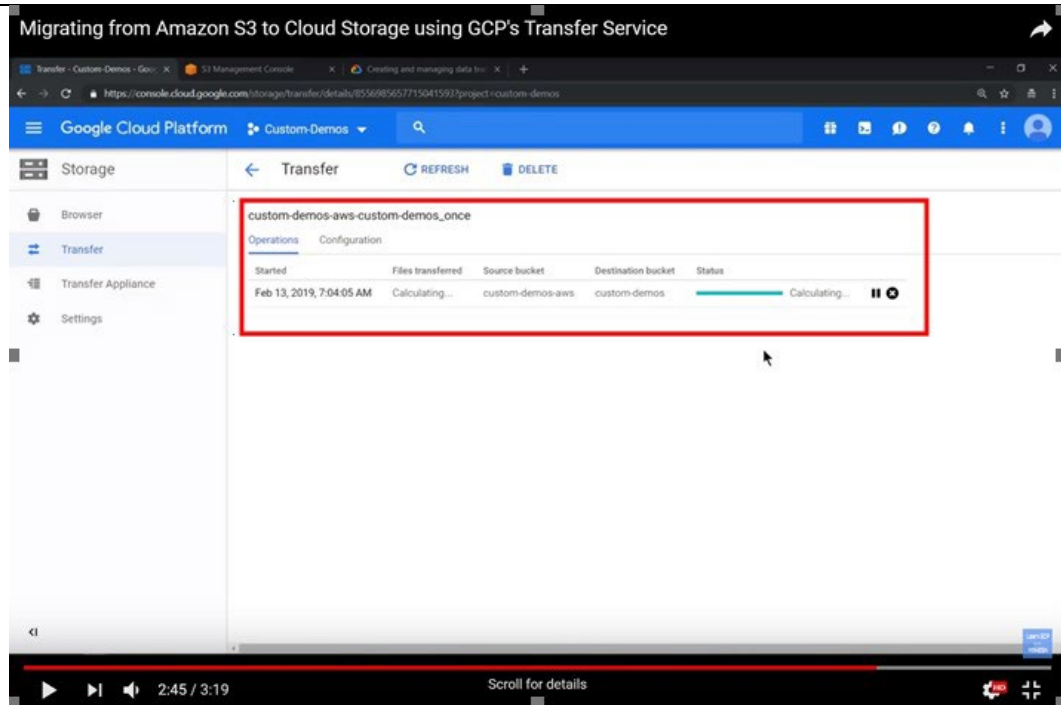
receiving first data via a network at a second interface;

Google Storage Transfer Service receive first data (data from online sources) via a network

	<div><h3>Complete large-scale data transfers fast</h3><p>Transfer petabytes of data from on-premises sources or other clouds over online networks—billions of files and 10s of Gbps. Optimize your network bandwidth and accelerate transfers with scale-out performance.</p></div> <div><h4>KEY FEATURES</h4><h3>Reliable and secure data transfer services</h3><h4>Storage Transfer Service for cloud data</h4><p>Transfer data from online sources like AWS S3 and Azure Blob Storage to Cloud Storage (/storage) in one simple process. Perform transfers as needed to enable a multicloud strategy, or periodically move data as part of a data-processing pipeline or analytical workflow.</p></div> <div><p>Storage Transfer Service Google Cloud</p><p>Source: https://cloud.google.com/storage-transfer-service#storage-transfer-service</p></div>
--	--



Cloud Storage as a data lake | Cloud Architecture Center | Google Cloud
 Source: <https://cloud.google.com/architecture/build-a-data-lake-on-gcp>



Source: <https://youtu.be/sBNtvG5D2CM?t=154>

storing second data at a cache memory;

During the execution of this service all data and commands received on google cloud platform are processed only after loading into fast memory (like cache) as evident from code of a Function Shown below, Hence, getting the files from Amazon S3 and printing file name in Google Cloud Platform bucket appears to relate to “second data” as claimed.

```
def list_gcs_objects(google_access_key_id, google_access_key_secret, bucket_name):
    """Lists GCS objects using boto3 SDK"""
    # Create a new client and do the following:
    # 1. Change the endpoint URL to use the
    #    Google Cloud Storage XML API endpoint.
    # 2. Use Cloud Storage HMAC Credentials.

    client = boto3.client(
        "s3",
        region_name="auto",
        endpoint_url="https://storage.googleapis.com",
        aws_access_key_id=google_access_key_id,
        aws_secret_access_key=google_access_key_secret,
    )

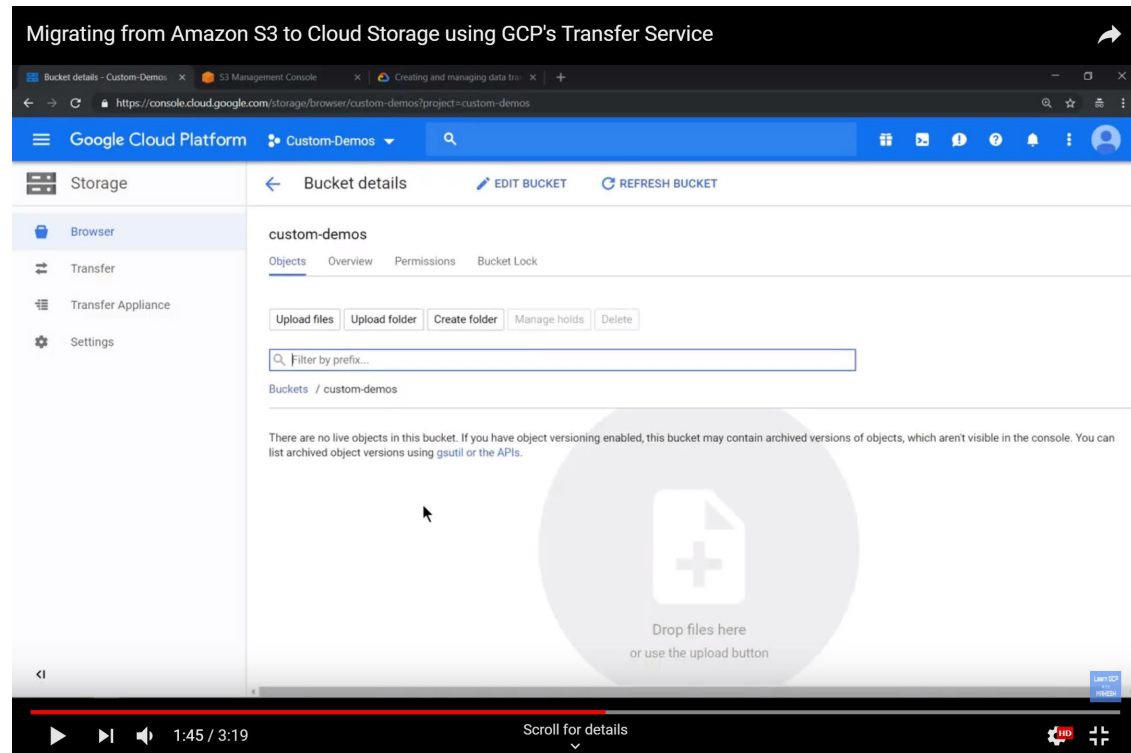
    # Call GCS to list objects in bucket_name
    response = client.list_objects(Bucket=bucket_name)

    # Print object names
    print("Objects:")
    for blob in response["Contents"]:
        print(blob["Key"])
```

Source: <https://cloud.google.com/storage/docs/migrating>

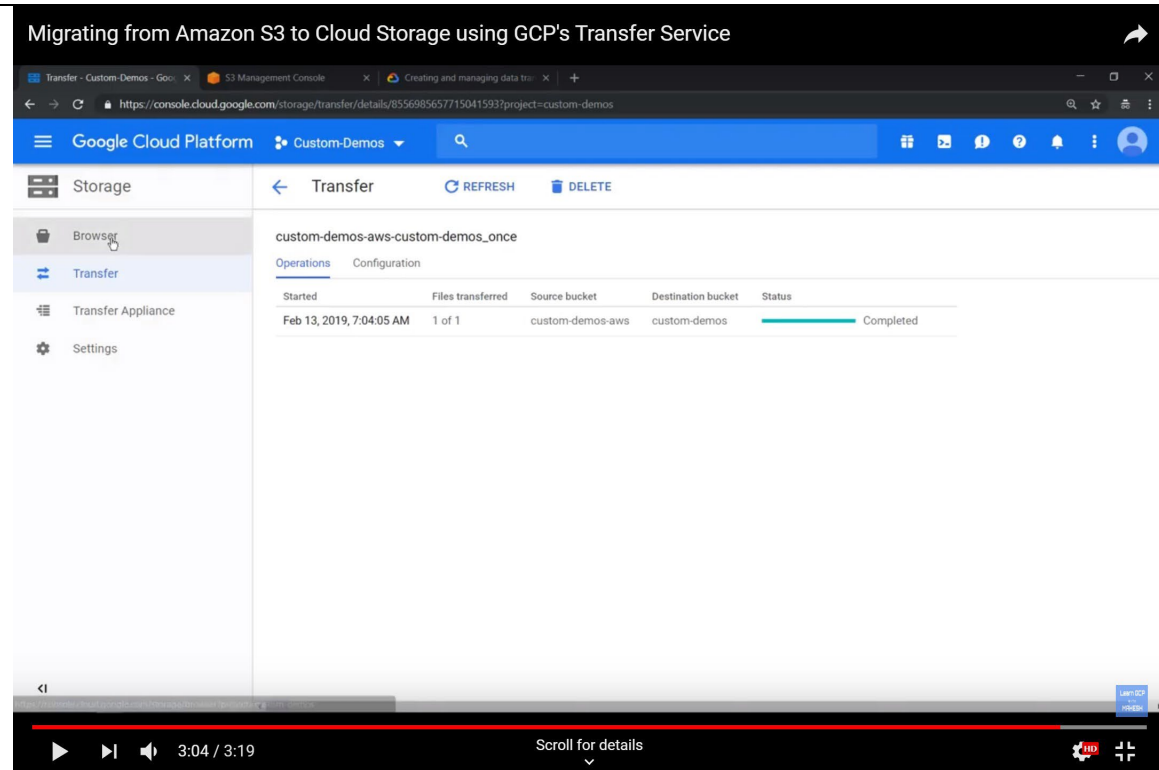
storing third data at a
storage device;

that a storage bucket is empty.



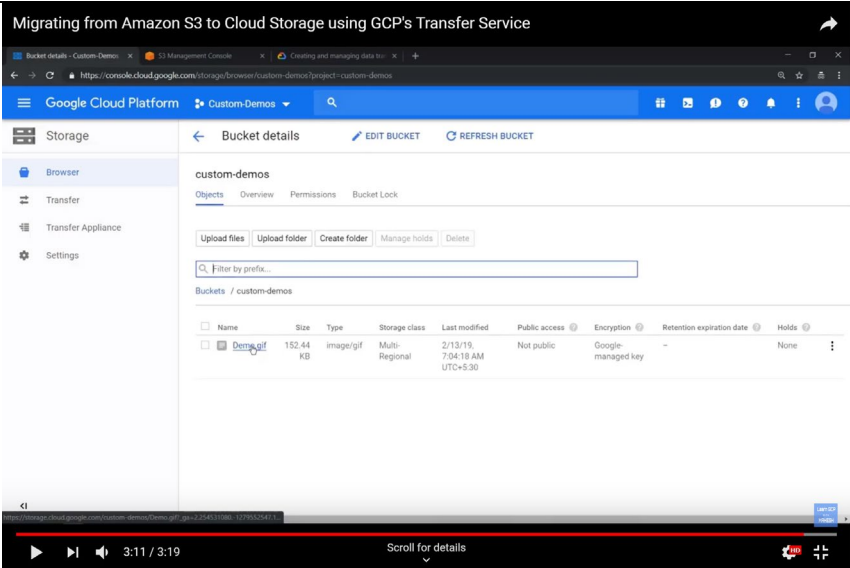
Source: <https://youtu.be/sBNtvG5D2CM?t=105>

the transfer process is complete.



Source: <https://youtu.be/sBNtvG5D2CM?t=105>

the data is stored in the storage bucket after completion of transfer process, which appears to relate to “storing third data at a storage device” as claimed.

	 <p>Source: https://youtu.be/sBNtvg5D2CM?t=191</p>
accessing the cache memory during processing of the I/O traffic; and	During the execution of this service all data and commands received on google cloud platform are processed only after loading into fast memory (like cache) as evident from codes I/O function in the follow evidence

	<pre>def list_gcs_objects(google_access_key_id, google_access_key_secret, bucket_name): """Lists GCS objects using boto3 SDK""" # Create a new client and do the following: # 1. Change the endpoint URL to use the # Google Cloud Storage XML API endpoint. # 2. Use Cloud Storage HMAC Credentials. client = boto3.client("s3", region_name="auto", endpoint_url="https://storage.googleapis.com", aws_access_key_id=google_access_key_id, aws_secret_access_key=google_access_key_secret,) # Call GCS to list objects in bucket_name response = client.list_objects(Bucket=bucket_name) # Print object names print("Objects:") for blob in response["Contents"]: print(blob["Key"])</pre> <p>Source: https://cloud.google.com/storage/docs/migrating</p>
<p>performing one or more access operations at the storage device based on the I/O traffic, the one or more access operations utilizing a</p>	<p>The request to transfer data from amazon's S3 is received via a dedicated channel and actual data from S3 is received at the google cloud storage via internet. Further the data received from S3 is loaded into cache memory and then written into cloud storage bucket. Therefore, it is evident that the path used for writing data into storage bucket is different (the Processor uses fast memory (i.e. cache memory) for processing the various functions which are used to complete the process of transfer of data from S3 to GCS)</p>

communication path between a processor and the storage device, the communication path distinct from the dedicated I/O channel.

```
def list_gcs_objects(google_access_key_id, google_access_key_secret, bucket_name):
    """Lists GCS objects using boto3 SDK"""
    # Create a new client and do the following:
    # 1. Change the endpoint URL to use the
    #    Google Cloud Storage XML API endpoint.
    # 2. Use Cloud Storage HMAC Credentials.

    client = boto3.client(
        "s3",
        region_name="auto",
        endpoint_url="https://storage.googleapis.com",
        aws_access_key_id=google_access_key_id,
        aws_secret_access_key=google_access_key_secret,
    )

    # Call GCS to list objects in bucket_name
    response = client.list_objects(Bucket=bucket_name)

    # Print object names
    print("Objects:")
    for blob in response["Contents"]:
        print(blob["Key"])
```

Source: <https://cloud.google.com/storage/docs/migrating>